Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Biomedical Informatics 39 (2006) 401–411
Journal of Biomedical Informatics
www.elsevier.com/locate/yjbin

Knowledge guided analysis of microarray data ☆

Zhuo Fang [a], Jiong Yang [c], Yixue Li [b], Qingming Luo [a], Lei Liu [b,d,*]

[a] Hubei Bioinformatics and Molecular Imaging Key Laboratory, Huazhong University of Science and Technology, Wuhan, Hubei 430074, PR China
[b] Shanghai Center for Bioinformatics Technology, Shanghai 200235, PR China
[c] Department of EECS, Case Western Reserve University, Cleveland, OH 44106, USA
[d] W.M. Keck Center for Comparative and Functional Genomics, University of Illinois at Urbana-Champaign, 1201 W. Gregory, IL 61801, USA

Received 27 April 2005
Available online 15 September 2005

Abstract

To microarray expression data analysis, it is well accepted that biological knowledge-guided clustering techniques show more advantages than pure mathematical techniques. In this paper, Gene Ontology is introduced to guide the clustering process, and thus a new algorithm capturing both expression pattern similarities and biological function similarities is developed. Our algorithm was validated on two well-known public data sets and the results were compared with some previous works. It is shown that our method has advantages in both the quality of clusters and the precision of biological annotations. Furthermore, the clustering results can be adjusted according to different stringency requirements. It is expected that our algorithm can be extended to other biological knowledge, for example, metabolic networks.

Keywords: Expression data; Gene ontology; Clustering

1. Introduction

DNA microarray technology permits parallel monitoring and large-scale screening of many genes simultaneously in the expression levels [1,2]. Analysis of microarray expression data is becoming one of the major bottlenecks in the utilization of this technology [3]. Among these analyses, clustering has been widely recognized as a useful data-mining tool for discovering similar patterns in expression dataset, which may lead to the insight of significant connections in gene regulatory networks.

So far, many traditional mathematical clustering methods have been applied in the context of clustering microarray expression data [4–11]. In these pure mathematical methods, genes are always clustered into disjoint groups, which cannot capture the biological fact that many gene products participate in more than one biological process [12]. Recently, some advanced mathematical algorithms, such as subspace clustering algorithms [13–16], were published, that may capture the coherence in a subset of genes and a subset of conditions. However, all of these algorithms only pay attention to mathematical similarity of genes and conditions, while the biological meaning of clusters is still neglected.

To overcome the above problem, biological knowledge is introduced in expression data analysis. The biological knowledge can be obtained from either scientific literatures or public databases, for example, gene functional annotation [17], MEDLINE database [18], metabolic networks [19], etc. Among them, gene ontology (GO), a large hierarchical vocabulary describing gene product functions in an organism-independent fashion, has been adopted by many recent researches [20,21]. Hvidsten et al. [22] published a systematic supervised learning approach to predict biological process. Liu et al. [20] also presented a novel technique by incorporating GO information into biclustering process. More than these, a large number of programs have been

☆ Availability: The source code of the program and the test data sets are available at: http://titan.biotec.uiuc.edu/clustering/.

* Corresponding author. Fax: +1 217 265 5066.
E-mail address: leiliu@uiuc.edu (L. Liu).

1532-0464/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.jbi.2005.08.004

developed to perform the statistical determination, interpretation and visualization of function profiles based on gene ontology, such as GOMiner [23], GOTree Machine [24], FunSpec [17], Onto-Tools [25], and GO::TermFinder [26]. All these methods and programs mainly focus on using knowledge of GO to evaluate or interpret clustering results, rather than attempting to improve clustering itself.

Recently, a co-clustering method, where GO is combined, was reported by Cheng et al. [27]. In Cheng's method, the GO term similarity and expression similarity are integrated together as similarity measurement in clustering, and thus the clusters get more stable and biologically meaningful. Different from Cheng's method, in this paper, we present a novel clustering algorithm, in which the expression profile is mapped to the modified structure of GO (GO tree). After that, the clustering process is based on the GO tree.

In the following sections, our method is demonstrated in details. In Section 2, the clustering algorithm is illustrated. In Sections 3 and 4, the results of our method on two real-world datasets [4] and some comparison with those of other published methods are presented. Finally, in Section 5, our method is discussed, and thus some conclusions are drawn. The paper ends with perspectives for other potential applications and suggestions for further improvements.

2. Methods

In our method, GO is introduced to guide the clustering of microarray gene expression dataset. There, firstly, a GO tree is constructed from GO data file. Subsequently, genes involved in the expression dataset are mapped to this GO tree via species related database, and unmapped nodes (terms) in GO tree are excluded. Thereafter, every node in this GO tree is checked from top to bottom. Genes mapped to a node and its descendant nodes form an initial cluster, and its expression similarity is calculated. If high expression similarity is obtained, the cluster is output, and the node and its descendant nodes are excluded from GO tree. Otherwise, no action is taken. After traversing the whole GO tree once, the algorithm produces a set of clusters. Refined by average trend constraint filtering, clusters with both high expression similarities and high function similarities are obtained. In the following sections our method is illustrated in details.

2.1. Construction of hierarchical tree using GO information

The gene ontology is a controlled vocabulary that can be applied to all organisms [28]. There are three categories in GO, namely molecular function, biological process and cellular component. Experimental studies show that among these three categories of GO, biological process agrees best with the hypothesis that similar expressions indicate a functional relation [29]. So we chose the category of biological process as our GO knowledge mapping base.

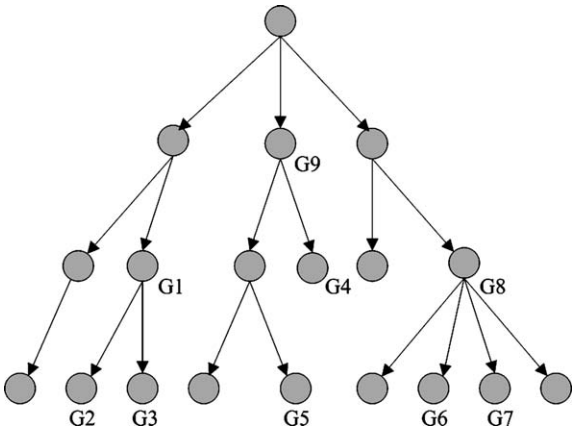

Fig. 1. The sketch map of GO tree with genes mapping on it. G_x represents the genes associated with the corresponding node for $x = 1, \ldots, 9$. The level of G_9 is 2. The level of G_1, G_4, and G_8 is 3 and the level G_2, G_3, G_5, G_6, and G_7 is 4. In this example, there are only nine nodes which are marked with genes.

GO hierarchy is originally described as a direct acyclic graph (DAG). For convenience, a flat file format GO is parsed and transformed into an ordered tree, a directed tree with an order defined for the children of every internal node of the tree, via the method of Lee et al. [30] As we know, a GO term may have more than one parent, that is, a GO term may have multiple paths from the root. To build an ordered tree, the occurrences of a same GO term in different paths are considered separately. The resulting graph is an ordered tree having GO terms as its nodes and the term 'biological_process' as its root. The information stored in each node includes: term name, term identifier and relationship with its parent term.

According to Lee's definition [30], a node is considered at level N of GO tree, $N = 1, 2, \ldots, H$ (H is the height of GO tree), if the depth of the node is N-1. For two GO nodes A and B such that (level of A) $= m$ and (level of B) $= n$ with $m < n$, we say that B is on a deeper level than A, or the level of B is greater than that of A. The sketch map of GO tree is shown in Fig. 1. After the GO tree is generated, all the following procedures will be based on the GO tree structure.

2.2. Mapping interested genes to GO tree

Only a part of the above GO tree, which consists of GO terms corresponding to genes in the to-be-clustered expression dataset, is interested. The GO term related to each gene in expression dataset can be obtained by searching the relevant database, for example, Saccharomyces Genome Database for yeast genes (http://www.yeastgenome.org/). All these obtained GO terms will be marked in the GO tree structure generated in Section 2.1. If one gene has more than one corresponding GO terms, all of them are marked. During clustering, only nodes that are marked will be considered, while all other nodes are neglected. The

mapping relationship between genes and GO terms is recorded, so that, in the following steps, the groups of GO terms can be inverse-mapped into gene clusters.

2.3. Clustering of genes

For a gene expression matrix (dataset), where each row represents a gene while each column represents a condition, we introduce *mean squared residue score* [15] to assess the expression correlation of a set of genes within a particular cluster. In reference [15], a cluster may consist of a subset of conditions, however, in our work, all conditions are considered during the clustering process. Therefore, the *mean squared residue score* of a cluster is redefined as following.

Definition 1. Given expression matrix (G, C), where G is the set of genes and C is the set of conditions, for a subset of genes $I \subset G$, the *mean squared residue score* (*msrs*) of the submatrix specified by (I, C) is defined as following:

$$H(I,C) = \frac{1}{|I||C|} \sum_{i \in I, j \in C} \left(a_{ij} - a_{iC} - a_{Ij} + a_{IC}\right)^2, \tag{1}$$

where a_{ij} is the element of the expression matrix, and

$$a_{iC} = \frac{1}{|C|} \sum_{j \in C} a_{ij}, \quad a_{Ij} = \frac{1}{|I|} \sum_{i \in I} a_{ij}, \tag{2}$$

$$a_{IC} = \frac{1}{|I||C|} \sum_{i \in I, j \in C} a_{ij} = \frac{1}{|I|} \sum_{i \in I} a_{iC} = \frac{1}{|C|} \sum_{j \in C} a_{Ij}. \tag{3}$$

Here a_{iC} is the average expression level of gene i across all conditions while a_{Ij} is the average expression of condition j for all genes in I. In addition, a_{IC} is the average expression of all genes in I across all conditions in C. This score captures the fluctuation level of gene expression data within a cluster. Low score indicates strong coherence in the cluster, for example, $msrs = 0$ means that the gene expression levels fluctuate in unison. The users can set an appropriate threshold δ of *msrs* to qualify the cluster. If the *msrs* of a cluster is below δ, the cluster will be considered as qualified. Otherwise, the patterns in this cluster are considered not coherent as well as we expected.

For a given δ, our clustering algorithm proceeds as follows. For every level of the GO tree, we start from the leftmost node of this level. For instance, in Fig. 1, we choose the node corresponding to G_9 in level 2 (only nodes mapped with genes will be considered in our process). If the node is marked with 'clustered,' we then go to the next node in the same level. Otherwise, all its descendant nodes will be selected, for example, nodes corresponding to G_4 and G_5 for the descendant nodes of G_9. Assuming that the set of genes which are mapped to these descendant nodes and the node itself is I, we denote the expression profiles of genes I as a matrix, B, specified by (I, C). In this example, I represents the set (G_9, G_4, G_5). Thereafter, we calculate the *msrs* value of B and compare it with δ. If the *msrs* value is below δ, this cluster will be output; subsequently, the algorithm goes to the next node in this level. All nodes contributing to this cluster will be marked with 'clustered.' Otherwise, no change of the marking in the tree will be made and we go to the next node in this level directly. In this example, the node corresponding to G_{10} is the only marked node of level 2. Thus we go to the nodes in level 3. This process repeats for all nodes in the level and then goes to the next level. The iteration terminates when every level of the GO tree has been visited. The detailed algorithm is listed as following:

ALGORITHM: *cluster_tree(H, TR, EM)*
INPUT: *H*: GO tree
TR: Threshold
EM: Expression matrix
OUTPUT: Gene clusters

FOR $i = 1$ to max level of H **DO**
  FOR $j = 1$ to max node index of level i **DO**
    IF the node j is clustered, continue
    ELSE
      Find all descendant nodes of j on H, with their corresponding gene set G.
      Find all genes corresponding to node j, calculate its union with G, obtain the gene set I
      Select the expression profiles of genes in I from EM and thus build a sub-matrix B
      Calculate the *msrs* value $MSRS$ of B
      IF $MSRS > TR$, continue
      ELSE output a cluster including all genes in set of I, and mark the node j and its descendant nodes with 'clustered'
      END IF
    END IF
  END FOR
END FOR

2.4. Average trend constraint filtering

Although *msrs* is used to ensure the coherence of a cluster, there are still some exceptional cases. For instance, let us assume that there is a cluster containing 20 genes. Two of these 20 genes do not have the similar fluctuation of gene expressions with others, while the expression patterns of the rest genes are very similar. In this case, the *msrs* value may be less than the threshold, because the inconsonance brought by two genes is not so sufficient to influence the coherence of the cluster. This kind of case is not what we want. Thus, the *average trend constraint filtering* is devised to remove the genes whose expressions are aberrant to the average trend of the cluster. The trend means the increase or the decrease of gene expression levels between two conditions. The average trend means the predominant trend between two conditions of all genes in the cluster. For

example, given a cluster consisting of genes {G1, G2, ..., G20} and conditions C1 and C2, if 12 of these 20 genes increase their expression level from C1 to C2 and 5 of them decrease and 3 of them remain un-change, the average trend of the cluster between C1 and C2 is "up." To a cluster output from the above clustering algorithm, we calculate the average trend for each adjacent pair of conditions and thus produce an average trend vector. Subsequently, after a threshold for the maximum tolerable number of inconsistent trends is defined, the trend vector of each gene in the cluster is compared with the average trend vector. If the difference between a gene's trend vector and the average trend vector exceed the threshold, the gene will be removed from the cluster.

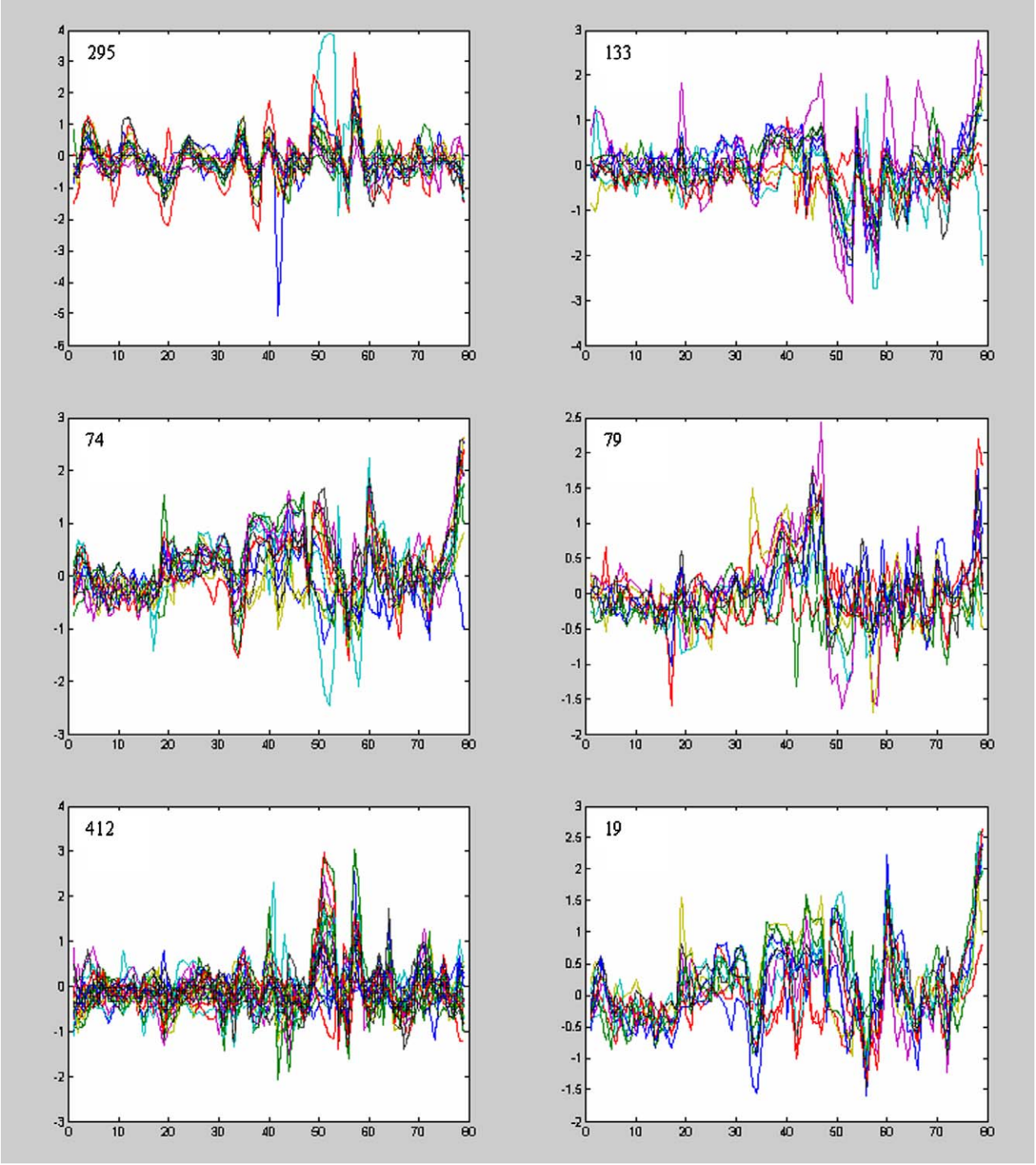

Fig. 2. Profiles of some clusters with the numbers indicate the orders of the clusters.

3. Experimental methods

3.1. Data preparation

To validate our algorithm, the well-known data set by Eisen et al. [4] was adopted. There, gene expression in the budding yeast *Saccharomyces cerevisiae* was studied during the diauxic shift, the mitotic cell division cycle, sporulation, and temperature and reducing shocks by using microarrays containing essentially every ORF from this fully sequenced organism. Each cell in the expression matrix represents the measured Cy5/Cy3 fluorescence ratio at the corresponding target element on the appropriate array. All ratio values were log transformed (base 2 for simplicity) to treat inductions or repressions of identical magnitude as numerically equal but with opposite sign. Using the hierarchical clustering methods, Eisen et al. [4] successfully clustered the gene expression profiles. We investigated the genes of Fig. 2 in Eisen et al., which can be accessed from http://genome-www.stanford.edu/clustering/. There are 2467 genes and 79 conditions. Values for missing data (1.9% of the data) were replaced with 0. The Saccharomyces Genome Database (SGD) downloaded from http://www.geneontology.org/ was used to extract GO term [31]. We used the SGD (Revision 1.923) and the GO (Revision 3.71) in our experiments. Only biological process of GO was discussed.

Since there are too many genes in expression matrixes and some of them do not show any significant fluctuation across different conditions, in this experiment, we selected genes whose expression profiles show significant variation among all conditions and removed those without significant fluctuations. Particularly, we chose genes whose expression profiles contain at least one value out of the range of $[-1,1]$. The range can be adjusted according to users' preference.

3.2. Determination of parameter

To obtain appropriate threshold value δ, we calculated the *msrs* values for each GO term corresponding to the genes in the input data set. Table 1 is a segment of this example result. From it, one can choose an appropriate threshold value based on his requirement. For example, if one needs to see a cluster derived from a particular GO term, he can find the *msrs* value of this term and set a threshold δ a little larger than the *msrs* value. In addition, we listed the level number of each GO term; therefore, user can determine the δ value by skimming to the *msrs* values of the terms on the wanted level.

The threshold in average trend constraint filtering is usually set as 30–40% of the condition numbers. Low threshold leads to consistent but small clusters. We used 30 as the maximum number of the tolerable incorrect trends.

3.3. Evaluation criteria

To assess the reliability of our result clusters with GO, a function *WR* is defined to evaluate the coherence of annotation. For a cluster C whose annotation space is A, we suppose a is the most frequently occurred annotation in A.

$$WR = 1 - \frac{cor\_num}{whl\_num}, \tag{4}$$

where *cor_num* is the number of genes annotated by a and *whl_num* is the total number of genes in C. Obviously, *WR* can measure the inconsistency of annotations in a cluster. Smaller *WR* implies stronger coherence.

3.4. Implementation

Our algorithm was implemented in C++. With the parameters specified above, clusters were discovered in less than 1 min using a PC with a 2.4 GHz CPU under Windows environment with 512 MB RAM. The source code of the program and the example data sets are available at http://titan.biotec.uiuc.edu/clustering/.

4. Results

Our algorithm was tested using the dataset prepared in Section 3.1. Setting 0.21 to δ, we obtained 510 clusters. After the *average trend constraint filtering*, 423 clusters remained. For each cluster, we used a GO term as the annotation to describe the biological function of the cluster. The profiles of some clusters are shown in Fig. 2. Some of their parameters are shown in Table 2.

Table 1
Segment example of *msrs* values for GO terms on various levels

Level	Term	GO id	*Msrs* value
0	Gene_Ontology	GO:0003673	0.320990
1	Biological_process	GO:0008150	0.320990
2	Cellular process	GO:0009987	0.316994
2	Development	GO:0007275	0.286936
2	Physiological process	GO:0007582	0.320997
2	Regulation of biological process	GO:0050789	0.283722
2	Viral life cycl	GO:0016032	0.000000
3	Cell communication	GO:0007154	0.270764
3	Cell differentiation	GO:0030154	0.316003
3	Cellular physiological process	GO:0050875	0.315899

0 *msrs* value means that there is no gene or only one gene corresponding to this term.

Table 2
Some parameters of clusters shown in Fig. 2

Cluster	Term	Size	*Msrs* value	WR
295	Mismatch repair	14	0.201853	0.285714
133	Group transfer coenzyme metabolism	12	0.205341	0.181818
74	ATP synthesis coupled electron transport	11	0.131158	0.000000
79	Protein complex assembly	10	0.183946	0.100000
41	Chromatin modification	24	0.207663	0.000000
19	Electron transport	14	0.200516	0.250000

It has been pointed out in previous section that one gene may be mapped to more than one GO terms, and these terms will be considered as separate nodes on the GO tree in our algorithm. According to this phenomenon, we permit the existence of overlapping clusters, that is, one gene may appear in several clusters. For instance, gene YAL016W belongs to cluster 14, cluster 153, cluster 165, cluster 369, and cluster 454. The biological terms of each cluster are bud growth (cluster 14), dephosphorylation (cluster 153), protein amino acid dephosphorylation (cluster 165), cell cycle checkpoint (cluster 369) and mitotic checkpoint (cluster 454). This means that gene YAL016W participates in all these processes. There are 810 genes in our experiment that were assigned to multiple clusters.

In following Section 4.1, we compare our clusters with those from Eisen's published method. It is shown that our method can get better result with the GO structure introduced and referred. Moreover, we also compare our annotation with a recent result published by Lee, who used GO to interpret the result of Eisen's clustering. The comparison given in Section 4.2 demonstrates that our annotations are more detailed. Besides, in Section 4.3, we discuss the dependence of thresholds versus meaningful clusters, also with some statistical value given.

4.1. Comparison of clustering results

We compared our results with that from Eisen's hierarchical clustering methods. There were many overlaps between our clusters and Eisen's clusters. Table 3 shows two examples. Our cluster 75 contains 8 of the 15 genes of Eisen's cluster 6. From the keywords of Eisen one can see that all these 8 genes are about ATP synthesis, while 6 of the remainder 7 genes are not annotated with ATP synthesis. In addition, four genes (YPL271W, YBR039W, YDL130W, and YGR008C) that were not included in Eisen's cluster 6, are included in our cluster and annotated as ATP synthesis. The similar phenomenon appears in Eisen's cluster 2 and our cluster 335. Twenty-four out of 27 genes in Eisen's cluster 2 are found in our cluster. Our cluster 335 includes 5 more genes that are annotated as Protein degradation, which are the main keywords of Eisen's cluster 2.The profiles of these clusters are shown in Fig. 3.

Table 4 shows the comparison of WR values between Eisen's clusters and ours. It can be seen that our results are much better than Eisens' in the aspect of coherence of annotation.

To further validate our method, we applied our method to another data set, the yeast *Saccharomyces cerevisiae* cell cycle data set of Cho et al. [32], which captures 6220 mRNA species in synchronized *S. cerevisiae* batch cultures through 17 time points. Tavazoie et al. [33] clustered the most variable 3000 ORFs into 30 clusters using *k*-means algorithm. We used the genes with identified SGD ORF names and selected 5000 as δ based on the range of the data set. 574 clusters were obtained and 513 clusters remained after *average trend constraint filtering*. The *WR* values from our clusters and Tavazoie's clusters are listed in Table 5. It indicates that the clusters we obtained are more coherent.

4.2. Comparison of annotation

We compare our results with Lee's [30] to demonstrate how the introducing of GO can lead to proper interpretations of clusters. Lee et al. utilized a graph algorithm to extract common biological attributes of the genes within a cluster based the GO hierarchy. They annotated the top 10 clusters of Eisen et al. [30] and obtained the representative biological meanings of each clusters by *AverPd*. Lee claimed that the clusters except 2, 7, and 9 had inconsistent functional contexts. The results are listed in Table 6. There are three observations.

(1) Some of Eisen's clusters are highly overlapped with our clusters, such as cluster 2 and 9. For these clusters, our annotation is the same as the Lee's annotation.
(2) Some of our clusters are the sub-clusters of Eisens', for example, cluster 1 and 7, and thus the associated annotations are also the child terms of Lee's annotations.
(3) There are some Eisen clusters, which are partitioned into a set of clusters in our results. For example, Eisen clusters 5 and 8 are annotated as 'protein biosynthesis' by Lee et al., while our method divided them into 12 clusters. The GO term of these sub-clusters are 1–4 levels deeper than 'protein biosynthesis,' respectively, and all have *msrs* values smaller than 0.281309.

Due to the fact that the term 'mRNA splicing' is obsolete, the annotation of cluster 3 is changed to 'mRNA processing.'

The above results demonstrate that our algorithm produced more specific annotation while compared with Lee's method.

Table 3
Comparison with Eisen's clusters

Gene names in Eisen's cluster 6	Eisen keywords	Gene names in our cluster 75	Gene names in Eisen's cluster 2	Eisen keywords	Gene names in our cluster335
YKL193C	Glucose repression		YFR004W	Transcription	
YGL187C	Oxidative phosphorylation		YGR048W	Protein degradation	YGR048W
YGL191W	Oxidative phosphorylation		YDR427W	Protein degradation	YDR427W
YLR395C	Oxidative phosphorylation		YKL145W	Protein degradation	YKL145W
YBL099W	ATP synthesis	YBL099W	YGL048C	Protein degradation	
YDR298C	ATP synthesis	YDR298C	YFR050C	Protein degradation	YFR050C
YJR121W	ATP synthesis	YJR121W	YDL097C	Protein degradation	YDL097C
YLR038C	Oxidative phosphorylation		YOR259C	Protein degradation	YOR259C
YPL078C	ATP synthesis	YPL078C	YPR108W	Protein degradation	YPR108W
YDR377W	ATP synthesis	YDR377W	YER021W	Protein degradation	YER021W
YLR295C	ATP synthesis	YLR295C	YGR253C	Protein degradation	YGR253C
YBR039W	ATP synthesis		YGL011C	Protein degradation	YGL011C
YDL004W	ATP synthesis	YDL004W	YMR314W	Protein degradation	YMR314W
YKL016C	ATP synthesis	YKL016C	YGR135W	Protein degradation	YGR135W
YJL166W	Oxidative phosphorylation		YER012W	Protein degradation	YER012W
	ATP synthesis	YPL271W	YPR103W	Protein degradation	YPR103W
	ATP synthesis	YBR039W	YJL001W	Protein degradation	YJL001W
	Protein synthesis	YDL130W	YOR362C	Protein degradation	YOR362C
	ATP synthesis	YGR008C	YOR157C	Protein degradation	YOR157C
			YOL038W	Protein degradation	YOL038W
			YBL041W	Protein degradation	YBL041W
			YHR200W	Protein degradation	YHR200W
			YDR394W	Protein degradation	YDR394W
			YOR117W	Protein degradation	YOR117W
			YFR052W	Protein degradation	
			YDL147W	Protein degradation	YDL147W
			YOR261C	Protein degradation	YOR261C
				Protein degradation	YDL007W
				Protein degradation	YDL020C
				Protein degradation	YER094C
				Protein synthesis	YGL017W
				TRNA processing	YIL075C
				Protein degradation	YML092C
				Protein degradation	YMR022W

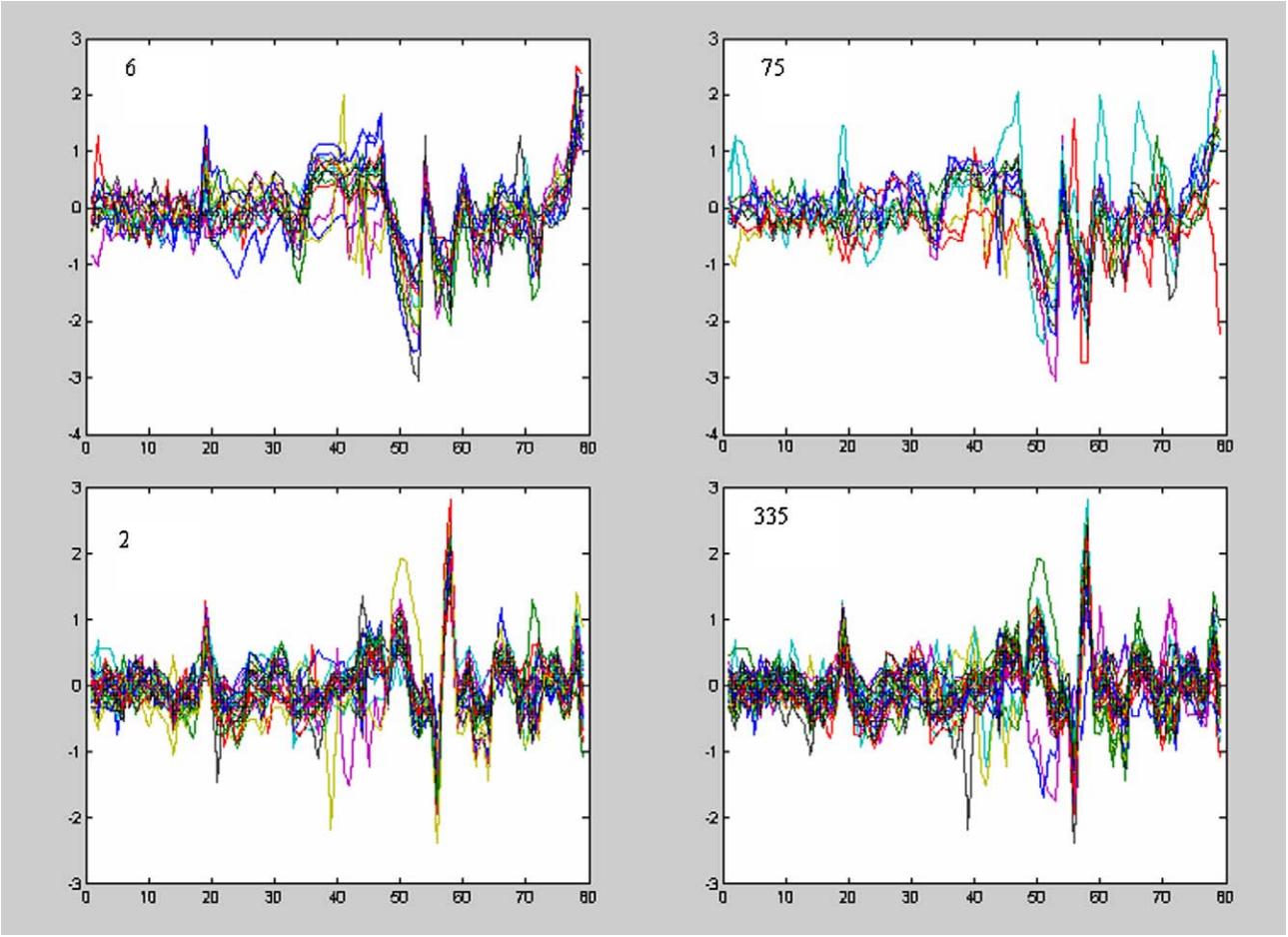

Fig. 3. Profiles of clusters: Eisen's cluster 6 (left up), our cluster 75 (right up), Eisen's cluster 2 (left down), our cluster 335 (right down).

Table 4
Comparison of *WR* values between Eisen's clusters and ours

Clusters	Number of clusters	Number of zero WR	Average WR
Eisen's	9	1	0.3508
Ours	423	258	0.1839

Table 5
Comparison of *WR* values between Tavazoie's clusters and ours

Clusters	Number of clusters	Number of zero WR	Average WR
Tavazoie's	30	0	0.2799
Ours	513	394	0.0905

4.3. Dependence of threshold versus meaningful clusters

As described in Section 2.1, the nodes on GO tree are labeled with different levels, and the nodes in deeper levels have more biologically meaning [34]. In this sense, the clusters with interpretations of deeper GO terms are expected to contain more accurate biological information and be more significant in future analysis. So, we investigate the relationship between msrs thresholds and annotation results. With the dataset mentioned in Section 3.1, the distributions of clusters over different GO tree levels for various thresholds are given in Fig. 4. It is shown that, for smaller threshold, the distribution is moved to deeper GO tree level, which means more specific annotation. The average levels of thresholds 0.15, 0.21, 0.25, and 0.3 are 7.37, 6.88, 6.66, and 6.19, respectively. We conclude that smaller threshold can lead to more specific clusters. Since there is a tradeoff between cluster size and its annotation level, we have to choose appropriate threshold to obtain reasonable size of clusters. In this sense, there is a tradeoff between the cluster size and annotation precision as well.

5. Discussion

There are many clustering methods for analysis of microarray expression data. Most of them are based on the similarity of the expression patterns. To further extract the biological meaning from clusters, some other algorithms and programs are developed to character clusters with GO terms [35,36,24]. Usually, these programs use statistical tests or topological properties to assess the quality of clustering results. In most of the cases, it is hard to interpret the clustering results, because some genes in the same cluster might have no biological similarity at all. Our method tries to overcome this problem via combining

Table 6
Comparison with Lee's results

Cluster	Lee's annotations (*msrs*)	Our annotations (*msrs*)
1	Microtubule nucleation (0.245740)	Microtubule nucleation—tubulin folding (0.121737)
2	Protein–ligand dependent (0.204963)	Protein–ligand dependent (0.204963)
3	mRNA splicing (0.185316)	mRNA processing (0.185316)
4	Glycolysis (0.313494)	Glycolysis (0.313494)
5 and 8	Protein biosynthesis (0.281309)	Protein biosynthesis—glycoprotein biosynthesis—protein amino acid glycosylation—O-linked glycosylation (0.195641) Protein biosynthesis—glycoprotein biosynthesis—protein amino acid glycosylation—N-linked glycosylation—dolichol-linked oligosaccharide biosynthesis (0.188391) Protein biosynthesis—glycoprotein biosynthesis—protein amino acid glycosylation—N-linked glycosylation—*N*-glycan processing (0.193912) Protein biosynthesis—glycoprotein biosynthesis—protein amino acid glycosylation—N-linked glycosylation—N-linked glycosylation via asparagines (0.030010) Protein biosynthesis—lipoprotein biosynthesis—protein lipidation (0.201316) Protein biosynthesis—mannoprotein biosynthesis (0.169101) Protein biosynthesis—regulation of protein biosynthesis—regulation of translation—negative regulation of translation (0.032459) Protein biosynthesis—regulation of protein biosynthesis—regulation of translation—regulation of translational elongation (0.137897) Protein biosynthesis—regulation of protein biosynthesis—regulation of translation—regulation of translational fidelity (0.154353) Protein biosynthesis—regulation of protein biosynthesis—regulation of translation—regulation of translational initiation (0.190178) Protein biosynthesis—translation—translation termination (0.163545) Protein biosynthesis—translation—tRNA amino acylation for protein translation (0.194888)
6	ATP synthesis coupled proton transport (0.192515)	ATP synthesis coupled proton transport (0.192515)
7	Chromatin assembly/disassembly (0.300438)	Chromatin assembly/disassembly—nucleosome assembly (0.145798)
9	DNA replication initiation (0.252383)	DNA replication initiation (0.252383)
10	Aerobic respiration (0.233765)	Aerobic respiration (0.233765)

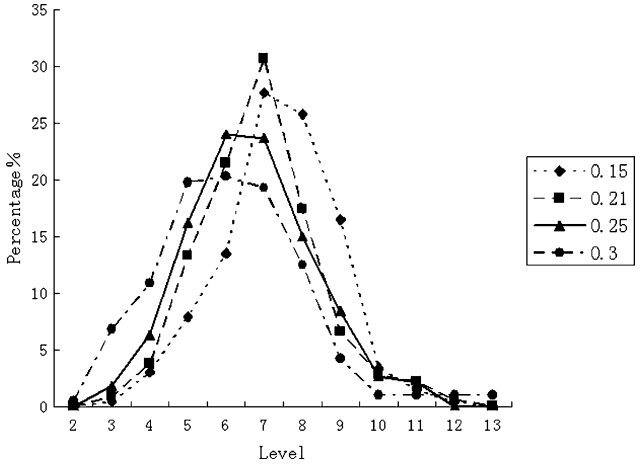

Fig. 4. Distributions of clusters for various thresholds.

gene expression similarity with function similarity. Gene Ontology is introduced as the prior function knowledge. Therefore, several advantages are obtained.

First, our clusters items are based on a similarity measure that depends on both expression profiles and biological functions, which are equally essential for gene clusters. These two are inseparable and exchangeable. It is the main difference between our method and others, where annotating is after clustering. As we know, generally, in microarray expression data analysis, clustering and annotation are undertaken separately, annotation after clustering. It is of high possibility that genes with dissimilar functions are clustered into one cluster because of their high expression similarity. This situation is what biologists are trying to obviate. In our clustering method, GO is referred amid clustering, therefore, to a cluster, the function similarity is guaranteed before expression similarity is obtained. Obviously, our clustering method will produce clusters with high similarities of both expression and function. As we have seen in the above experimental results, our clustering method brought higher coherent clusters with specific biological meanings at the same time. Moreover, as clustering and annotation are undertaken at one time, the user's operation is facilitated.

Second, the similarity function used in our clustering (*msrs*) depends on a context, which is best defined as a subset of the attributes. Compared to the commonly used methods, for example, Euclidean distance and Pearson's correlation coefficient, *msrs* can valuate the fluctuation coherence among multi-genes while the other two can only measure the distance between two genes (Euclidean distance) or

the similarity of expression patterns between two genes (Pearson's correlation coefficient). Therefore, it shows more reasonability of *msrs* to get correlative gene sets.

Third, our method allows genes to be included in multiple clusters, and thus allow one gene to be associated with more than one function categories. That is consistent with the well-known fact that many gene products participate in more than one biological process. Obviously, it reflects the reality in the functionality of genes.

Our method can be applied in several usages. First, given a microarray dataset and a biological function, it can be used to find out the corresponding cluster of genes. We find the corresponding GO term by given function, and then obtain the *msrs* value of this term. Thereafter, the desired cluster can be calculated by setting the threshold a little larger than the *msrs* values. Second, it can satisfy the requirement of different annotation stringencies. If we want to get clusters whose annotation GO terms are below level *n* in the GO tree, we can find all the *msrs* values of terms at level *n* and set a threshold equal to the minimum value of them. The results will be clusters with more precise terms than level *n*.

Moreover, our algorithm can be easily generalized. In this paper, we have already used GO to demonstrate how a particular biological knowledge is integrated in analysis of microarray data. It can be imagined that, other biological knowledge, for example, pathways, can be used instead to create the initial cluster. In this sense, this methodology may be further applied to any kinds of analysis on other biological entities.

At the end, it should be pointed out that there are some improvements needed in further researches. For example, for genes corresponding to high level GO terms, the *msrs* values are quite big. As a result, these genes will not be included in any clusters. Besides, our method depends strongly on the accurateness and completeness of the GO hierarchy. It can be expected that if the quality of GO get improved with more updates and knowledge accumulation, our method will bring better result.

References

[1] Chen JJ, Wu R, Yang PC, Huang JY, Sher YP, Han MH, et al. Profiling expression patterns and isolating differentially expressed genes by cDNA microarray system with colorimetry detection. Genomics 1998;51:313–24.

[2] Iyer VR, Eisen MB, Ross DT, Schuler G, Moore T, Lee JCF, et al. The transcriptional program in the response of human fibroblasts to serum. Science 1999;283:83–7.

[3] Yang J, Wang H, Wang W, Yu P. Enhanced biclustering on expression data. BIBE 2003.

[4] Eisen MB, Spellman PT, Brown PO, Botstein D. Cluster analysis and display of genome-wide expression patterns. Proc Natl Acad Sci 1998;95(25):14863–8.

[5] Herwig R, Poustka A, Muller C, Bull C, Lehrach H, O'Brien J. Large-scale clustering of cDNA-fingerprinting data. Genome Res 1999;9:1093–105.

[6] Heyer LJ, Kruglyak S, Yooseph S. Exploring expression data: identification and analysis of coexpressed genes. Genome Res 1999;9(11):1106–15.

[7] Smet FD, Mathys J, Marchal K, Thijs G, Moor BD, Moreau Y. Adaptive quality-based clustering of gene expression profiles. Bioinformatics 2002;18:735–46.

[8] Herrero J, Valencia A, Dopazo J. A hierarchical unsupervised growing neural network for clustering gene expression patterns. Bioinformatics 2001;17:126–36.

[9] Shamir R, Sharan R. Click: a clustering algorithm for gene expression analysis. In Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology (ISMB) 2000.

[10] Yeung KY, Fraley C, Murua A, Raftery AE, Ruzz WL. Model-based clustering and data transformations for gene expression data. Bioinformatics 2001;17:977–87.

[11] Jiang D, Pei J and Zhang A. DHC: a Density-based Hierarchical Clustering Method for Timeseries Gene Expression Data. In Proceeding of BIBE2003: 3rd IEEE International Symposium on Bioinformatics and Bioengineering 2003.

[12] Hvidsten TR, Lagreid A, Komorowski J. Learning rule-based models of biological process from gene expression time profiles using Gene Ontology. Bioinformatics 2003;19:1116–23.

[13] Getz G, Levine E, Domany E. Coupled two-way clustering analysis of gene microarray data. Proc Natl Acad Sci USA 2000;97(22):12079–84.

[14] Lazzeroni L, Owen A. Plaid models for gene expression data. Stat Sin 2002;12(1):61–86.

[15] Cheng Y, Church GM. Biclustering of expression data. In Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology (ISMB) 2000; 8:93–103.

[16] Yang J, Wang W, Wang H, Yu PS. δ-cluster: Capturing Subspace Correlation in a Large Data Set. In Proceedings of 18th International Conference on Data Engineering (ICDE 2002); 517–528.

[17] Robinson MD, Grigull J, Mohammad N, Hughes TR. FunSpec: a web-based cluster interpreter for yeast. BMC Bioinformatics 2002;35.

[18] Masys DR, Welsh JB, Fink JL, Gribskov M, Klacansky I, Corbeil J. Use of keyword hierarchies to interpret gene expression patterns. Bioinformatics 2001;17:319–26.

[19] Hanisch D, Zien A, Zimmer R, Lengauer T. Co-clustering of biological networks and gene expression data. Bioinformatics 2002;18:145–54.

[20] Liu J, Wang W, Yang J. Gene ontology friendly biclustering of expression profiles. CSB 2004.

[21] Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, et al. Gene ontology: tool for the unification of biology. Gene Ontol Consortium Nat Genet 2000;25:25–9.

[22] Hvidsten TR, Komorowski J, Sandvik AK, Lægreid A. Predicting gene function from gene expressions and ontologies. Pacific Symposium on Biocomputing World Scientific, Mauna Lani, HI 2001; p. 299–310.

[23] Zeeberg BR, Feng W, Wang G, Wang MD, Fojo AT, Sunshine M, et al. GoMiner: a resource for biological interpretation of genomic and proteomic data. Genome Biol 2003;4:R28.

[24] Bing Z, Denise S, Stefan K, Jay S. GOTree Machine (GOTM): a web-based platform for interpreting sets of interesting genes using Gene Ontology hierarchies. BMC Bioinformatics 2004;5:16.

[25] Sorin D, Purvesh K, Pratik B, Abhik S, Stephen AK, Michael AT. Onto-Tools, the toolkit of the modern biologist: Onto-express, onto-compare, onto-design and onto-translate. Nucleic Acids Res 2003;31:3775–81.

[26] Boyle EI, Weng S, Gollub J, Jin H, Botsterin D, Cherry JM, Sherlock G: GO::TermFinder—open source software for accessing Gene Ontology information and finding significantly enriched Gene Ontology terms associated with a list of genes. Bioinformatics Advance Access published August 5, 2004.

[27] Cheng J, Cline M, Martin J, Finkelstein D, Awad T, Kulp D, et al. A knowledge-based algorithm driven by Gene Ontology. J Biopharmaceut Stat 2004;14:687–700.

[28] Ashburner M, Ball CA, et al. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nat Genet 2000;25(1):25–9.
[29] Brown MPS, Grundy WN, Cristianini N, Sugnet CW, Furey TS, Ares M, et al. Knowledge-based analysis of microarray gene expression data by using support vector machines. Proc Natl Acad Sci USA 2000;97:262–7.
[30] Lee SG, Hur JU, Kim YS. A graph-theoretic modeling on GO space for biological interpretation of gene clusters. Bioinformatics 2004;20:381–8.
[31] Dwight SS, Harris MA, Dolinski K, Ball CA, Binkley G, Christie KR, et al. Saccharomyces Genome Database (SGD) provides secondary gene annotation using the GeneOntology (GO). Nucleic Acids Res 2002;30: 69–72.
[32] Cho RJ, Campbell MJ, Winzeler EA, Steinmetz L, Conway A, Wodicka L, et al. A genome-wide transcriptional analysis of the mitotic cell cycle. Mol Cell 1998;2(1):65–73.
[33] Tavazoie S, Hughes D, Campbell MJ, Cho RJ, Church GM. Systematic determination of genetic network architecture. Nat Genet 1999:281–5.
[34] King RD, Karwath A, Clare A, Dehaspe L. The utility of different representations of protein sequence for predicting functional class. Bioinformatics 2001;17:445–54.
[35] Doniger SW, Salomonis N, Dahlquist KD, Vranizan K, Lawlor SC, Conklin BR. MAPPFinder: using Gene Ontology and GenMAPP to create a global gene-expression profile from microarray data. Genome Biol 2003;4:R7.
[36] Khatri P, Draghici S, Ostermeier GC, Krawetz SA. Profiling gene expression using Onto-Express. Genomics 2002;79:266–70.